

REMARKS

This amendment responds to the second office action. Claims 71-74, 77, 78, 91-97 and 99-100 have been amended and claim 76 has been deleted.

As requested, applicants have updated the status of the parent application. Also pursuant to the Examiner's request, the abstract has been amended to make it more aptly descriptive of the invention. A new title and summary of the invention have also been provided to reflect the claimed invention.

The Examiner has rejected claims 71-90, 91, and 97 under 35 U.S.C. § 112 as being indefinite. The Examiner asserted that it was not recited that the instruction register could hold operands. This is due to a matter of semantics. While the term "instruction" includes the entire encoding (op-code and operand), it is often also used to refer to just the op-code. Similarly, op-code can refer to just the encoding of the functional descriptor (as in the preceding sentence), but it commonly also refers to the entire encoding. Hence, the pending claims 71, 91, and 97 now explicitly refer to the instruction register containing operands, instructions, or both. Applicant believes that this change also clarifies the scope problems referred to by the Examiner in claims 77-79, 81, 88, and 98-99. The text of claim 92 has been changed to more clearly define how instructions are skipped. On the basis of the above changes to the claims and these remarks, the rejection under § 112 is believed to be overcome.

The Examiner has rejected claims 71-100 under 35 U.S.C. § 103(a) as being unpatentable over Boufarah in view of May. To more clearly distinguish the instant case over the prior art, text to the effect that:

...one or more sequential instructions including at least one instruction that accesses operands or instructions or both located relative to said instruction groups...

has been added to independent claims 71 and 91. Neither Boufarah nor May teach that the operand and instruction references made by instructions are located with regard to the instruction group, rather than the executing instruction. In the microprocessor of this invention, for example, SKIPs are not a given number of instructions, but to the next instruction group, MICROLOOPs are not a given number of instructions, but back to the start of the current instruction group, BRANCHes do not use a given number of operand offset bits, but use the bits

remaining in the instruction group, and are to a destination at the beginning of an instruction group. Some of these examples are discussed more specifically below. Applicants believe the above changes in the pending independent claims 71, 91 and 97, make these claims patentably distinguishable over Boufarah in view of May, and all known prior art, and thus believes that the rejection of these claims has been overcome.

The advantages of accessing operands and instructions relative to instruction groups are several and fall into three groups.

First, the bottleneck in most computer systems is the memory bus (see page 10, line 3-7 of the application). By increasing the instruction bandwidth available over the bus the processor can either a) provide the same performance with lower cost memories (page 10, line 30 to page 11, line 4), or b) provide better performance with more expensive memories. Instruction bandwidth is a matter of reducing the number of instruction bits required to perform the same operation. The stack architecture of the processor promotes this and the processor instructions afforded protection by the pending claims attempt to further reduce the number of op-code bits required. SKIP supplies short branches and MICROLOOP supplies short loops with minimal opcodes. They do this by assuming the next or current instruction group as the destination rather than using additional opcode bits for a destination address. FETCH-VIA-PC (32-bit, page 55, lines 5-25) and LOAD-SHORT-LITERAL (8-bit, page 59, lines 6-33) supply two literal value sizes to minimize wasted bits when storing immediate data. BRANCHES of various types use a variable length operand (page 43, lines 16-24) to use only as many bits as are convenient. The branch address operands also address instruction groups (32-bit machine words) rather than bytes. This allows 4 times the branch distance available from a given number of branch operand bits, thus allowing shorter versions of the branches to be adequate. These compare to typical RISC processors where all instructions are 32-bits regardless of the amount of information needed to actually describe the operation to be performed.

Second, obtaining operands from the instruction stream can be complex and time consuming if the instructions are not all fixed in length. CISC machines (like a Pentium) must determine which bits, within an arbitrary stream of bytes from memory, contain operand bits for the current instruction, and take those bits and

then move them some amount to the right to align the least significant operand bit with the lowest bit on the CPU internal data bus. RISC machines typically have fixed length instructions to solve this problem by always placing the operands in the same bit areas in the instructions (thus moving the bits a fixed amount), but pay the penalty in instruction bandwidth, as described above. The processor in the instant case solves this problem by placing the operands always in the same place within an instruction group so they are in a fixed place. This gives the advantage of simple, fast and efficient access to operands, without the penalty of long instructions. By essentially placing operands and flowing instructions around them, the better code density of short instructions is achieved, and the implementation simplicity of fixed location operands is also achieved.

Third, instruction simplicity reduces the logic required and speeds execution. The processor, for instance, already "knows" how to fetch and execute instruction groups. Implementing SKIP, then, is simply a matter of directing execution to the next instruction group, which the processor already knows how to do. Similarly, for MICROLOOP, the processor already knows how to execute the current instruction group, and simply restarts this process until the count expires. With the addition of one set of conditional logic for SKIPS (to have a full set of conditional branches, page 46, lines 24-26), the processor knows "conditions" and these and other instructions easily become conditional. Again, the same applies to FETCH-VIA-PC, since the processor already knows how to fetch instruction groups, using the next instruction group as a 32-bits operand is easy. And also with LOAD-SHORT-LITERAL, since for branches the processor already knows right-justified operands. These various features synergistically work together to improve performance and reduce processor cost.

Turning to the rejections of certain of the dependent claims, at page 4 of the Office Action, the examiner asserts that May teaches SKIP at the top of column 16, but this is not the case. Examination of the location of the text "SKIP" at the top of column 16 shows that it is a continuation of the text from the bottom of column 15 for the instruction "jump non zero". The text "SKIP" in May in this context is used as descriptive procedural nomenclature as defined in May on column 10 line 61 thus: "The process SKIP terminates with no effect." However, even if the Examiner is correct in asserting that May teaches a SKIP, this does not negate patentability of the

claims as amended because Applicants do not contend that the concept of a "SKIP" is itself unique, as described on page 30, lines 6-9 of this application:

"Other machines (such as the PDP-8 and Data General Nova) provide the ability to skip a single instruction. The microprocessor 50 provides the ability to skip up to three instructions."

Thus, Applicants submit that in the instant case of SKIP, the behavior of SKIP within instruction groups is unique. All known to applicants previous cases of SKIP instructions have skipped a fixed number of instructions, specifically one instruction, as in the cited systems. In the instant case, SKIP skips from zero to three instructions remaining in the instruction group, the number dependent on where the SKIP is located among the four instructions in the instruction group. This feature of the invention is described at page 29, lines 24-35:

"...The SKIP instruction can be located in any of the four byte positions 420 in the 32-bit instruction register 108. If the test is successful [referring to previously discussed conditions on skips], SKIP will jump over the remaining one, two, or three 8-bit instructions in the instruction register 108 and cause the next four instruction group to be loaded into register 108. As shown, the SKIP operation is implemented by resetting the 2-bit microinstruction counter 180 to zero on line 422 and simultaneously latching the next instruction group into register 108. Any instructions following the SKIP in the instruction register are overwritten by the new instructions and not executed. ..."

and at page 46, lines 21-24:

"... SKIP instructions in the microprocessor skip any remaining instructions in a 4-byte instruction group and cause a memory fetch to get the next 4-byte instruction group."

Applicants submit that amended claims 72, 73, 92 and 93 are patentably distinguishable over all known prior art, and thus believes that the rejection of these claims has been overcome with an independent basis for patentability as argued in these remarks.

At page 4 of the Office Action, the examiner asserts that May teaches a system for executing loop instructions, but this is not the case. The referenced text refers to a "jump" instruction and lists uses of a "jump" instruction, two of which are "providing loops, exits from loops". However, the instant case refers to the

conventional definition of a loop instruction which includes a loop counter (a counted loop), something not contemplated by "jump" in the referenced art. Applicants do not content that concept of a counted loop is unique as such. Applicants submit that in the instant case of MICROLOOP, the behavior of MICROLOOP within instruction groups is unique. All known to applicants previous cases of counted loop instructions have required a destination address as an operand within the counted loop instruction. In the instant case, the destination address is assumed to be the first instruction of the current instruction group and is not stored as an operand within the instruction. Thus for any location of a MICROLOOP instruction within a four-byte instruction group, the destination address is the same. Subsequently, in addition to the MICROLOOP instruction, the loop can contain zero, one, two, or three instructions. This feature of the invention is described at page 30, lines 17-28 of the application:

"...MICROLOOP may be placed in the first, second, third or last byte 420 of the instruction register 108. If placed in the first position, execution will just create a delay equal to the number stored in the LOOP COUNTER 92 times the machine cycle. If placed in the second, third, or last byte 420, when the MICROLOOP instruction is executed, ... the 2-bit microinstruction counter is cleared, causing the preceding instructions in the instruction register to be executed again."

Applicants submit that the amended claims 74, 75, 94, and 95 are patentably distinguishable over all known prior art, and thus believes that the rejection of these claims has been overcome with an independent basis for patentability as argued in these remarks.

The examiner has rejected claims 88-90 as being unpatentable over May. While both May and the instant case involve variable length operands, May does not contemplate the variable length operand technique contemplated by the pending claims. May teaches the building of variable length operands four bits at a time by successively executing instructions that each supply an additional four bits. Building the maximum 16-bit operand requires executing at least four instructions. Other processors require changes to mode bits in the op-code to specify the length of the variable length operand. In the instant case, a single op-code is used and the width of a variable length operand depends on where the instruction begins in the current

instruction group. The operand for the instruction includes the bits from the op-code to the last bit in the current instruction group. This feature of the invention is described at page 33, lines 8-28:

“The microprocessor 50 handles operands of 8, 16, or 24 bits using the same op-code. Figure 20 shows the 32-bit instruction register 108 and the 2-bit microinstruction register 180 which selects the 8-bit instruction. Two classes of microprocessor 50 instructions can be greater than 8-bits, JUMP class and IMMEDIATE. A JUMP or IMMEDIATE op-code is 8-bits, but the operand can be 8, 16, or 24 bits long. This magic is possible because operands must be right justified in the instruction register. This means that the least significant bit of the operand is always located in the least significant bit of the instruction register. The microinstruction counter 180 selects which 8-bit instruction to execute. If a JUMP or IMMEDIATE instruction is decoded, the state of the 2-bit microinstruction counter selects the required 8, 16, or 24 bit operand onto the address or data bus. The unselected 8-bit bytes are loaded with zeros by operation of the decoder 440 and gates 442. The advantage of this technique is the saving of a number of op codes required to specify the different operand sizes in other processors.”

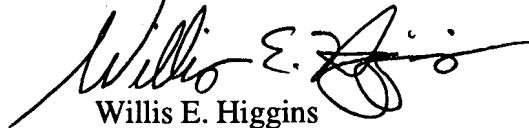
Applicants submit that claims 88-90 are patentably distinguishable over all known prior art, and thus believes that the rejection of these claims has been overcome with an independent basis for patentability as argued in these remarks.

Since the transitional GATT provisions are applicable to this application, Applicants are paying the fee to remove the finality of the action. A petition and fee for a three month extension of time is also enclosed. Based on the above changes to the claims and remarks, all of the claims in the application are believed to be

patentable over the prior art. This application is believed to be in condition for allowance, and allowance is solicited. Applicants propose to contact the Examiner after the filing of this amendment to see if there are any issues requiring further clarification.

Respectfully submitted,

COOLEY GODWARD LLP

A handwritten signature in black ink, appearing to read "Willis E. Higgins", with a stylized flourish at the end.

Willis E. Higgins
Reg. No. 23,025

Five Palo Alto Square
Palo Alto, CA 94306-2155
Telephone: (415) 843-5145